# Least-Squares Polynomial Filters for Ill-Conditioned Linear Systems

Jocelyne Erhel, Frédéric Guyomarc'H, Yousef Saad

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

# *Least-squares polynomial filters for ill-conditioned linear systems*

Jocelyne Erhel , Frédéric Guyomarc'h , Yousef Saad

**N°4175**

Mai 2001

THÈME 4


ISSN 0249-6399 ISRN INRIA/RR--4175--FR+ENG

INRIA
RENNES

# Least-squares polynomial filters for ill-conditioned linear systems

Jocelyne Erhel * , Frédéric Guyomarc'h † , Yousef Saad ‡

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet ALADIN

Rapport de recherche n ° 4175 — Mai 2001 — 28 pages

**Abstract:** An important problem which arises in several applications is to find the solution of an ill-conditioned Symmetric Semi-Positive Definite linear system whose right-hand side is perturbed by noise. In this situation, it is desirable for the solution to be accurate in the directions of eigenvectors associated with large eigenvalues, and to have small components in the space associated with smallest eigenvalues. A method is presented to satisfy these requirements, which is based on constructing a polynomial filter using least-squares techniques.

**Key-words:** Polynomial filters, least-squares polynomials, ill-conditioned systems, regularization, Tychonov regularization, image recovery, Chebyshev bases.

*(Résumé : tsvp)*

Work supported by the US Army Research Office under grant DA/DAAD19-00-1-0485 and by the Minnesota Supercomputer Institute.

* IRISA/INRIA, UR Rennes, Jocelyne.Erhel@inria.fr, http://www.irisa.fr/aladin/perso/erhel.html
† IRISA/INRIA, UR Rennes, Frederic.Guyomarch@inria.fr, http://www.irisa.fr/aladin/perso/fguyomar.html
‡ University of Minnesota, Department of Computer Science and Engineering, http://www-users.cs.umn.edu/ saad/

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

# Filtres polynomiaux pour des systèmes linéaires mal conditionnés

**Résumé :** Un problème important qui se pose dans plusieurs applications est de trouver la solution d'un système linéaire symétrique semi-défini positif et mal conditionné, avec un second membre perturbé par du bruit. Dans ce cas, il faut que la solution soit précise dans les directions propres associées aux grandes valeurs propres, qu'elle ait de petites composantes dans le sous-espace associé aux petites valeurs propres. Nous présentons une méthode qui satisfait ces conditions et qui est basée sur la construction d'un filtre polynomial solution d'un problème aux moindres carrés.

**Mots-clé :** Filtres polynomiaux, moindres carrés, systèmes mal conditionnés, régularisation, régularisation de Tychonov, restauration d'images, bases de Chebyshev.

**1. Introduction.** In several important applications it is required to solve a linear system of dimension $n$ the form

$$Ax = b \tag{1.1}$$

in which the matrix $A$ is very ill-conditioned and where the right-hand side $b$ is perturbed with noise. It is common that $A$ is singular and has a large number of singular values close to zero. As a result, the noise that is present in $b$ will tend to be amplified in the pseudo-inverse solution $x^\dagger = A^\dagger b$ of the system. Often the resulting solution becomes worthless as it is dominated by noise. Examples of typical applications of this nature are in discrete inverse problems for image recovery and in tomography.

In image recovery the matrix $A$ represents the blurring operator and is symmetric semi-positive definite. A number of 'regularization' strategies have been designed to recover a solution which is deblurred and filtered out of noise. The general principle of all regularization methods is to solve the system accurately only in the singular space associated with the large singular values while removing or reducing components associated with the small singular values since these are typically dominated by noise. Two prototypes of these methods are the Truncated Singular Value (TSVD) technique, and Tychonov regularization. These are briefly discussed in the next subsection. For references on TSVD and Tychonov regularization see [3].

In what follows we discuss regularization for the case when $A$ is symmetric semi-positive definite. Regularization can be viewed as a method for computing a filtered solution of the form

$$x_\phi = A^\dagger \phi(A) b \tag{1.2}$$

where the role of the filter function $\phi$ is to remove or dampen all eigen-components close to zero from the right-hand-side. This is done in order to prevent these components, which are typically dominated by noisy data, from being amplified.

Denoting by $H(t)$ the Heaviside function which is equal to zero for $t < 0$ and to one for $t \geq 0$, the filter function for the case of the truncated SVD regularization, can be written as

$$\phi_\epsilon(t) = H(t - \epsilon) \tag{1.3}$$

where $\epsilon$ is a "truncation" parameter. To apply the above filter function, a singular value decomposition is normally required.

In Tychonov regularization, the filter function is of the form

$$\phi_\rho(t) = 1 - \frac{1}{1 + (t/\rho)^2} = \frac{\rho^2}{\rho^2 + t^2} \tag{1.4}$$

where $\rho$ is a parameter. It is common in this case to use the conjugate gradient algorithm to compute an approximation to (1.2).

In [1], Calvetti et al. propose a method which uses an "exponential" filter of the form

$$\phi(t) = 1 - \exp\left[-\ (\rho/t)^2\right] \tag{1.5}$$

The goal is to obtain a function that is close to zero for $t$ near the origin and close to one when $t$ is far away from the origin. Clearly this function is not economically useable directly for computing an approximation of the form (1.2). Calvetti et., propose instead to expand this function in a basis of Chebyshev polynomials.

In this paper we propose a different type of filter. Our main motivation is to provide a sequence of polynomials that directly approximate a given ideal filter. Our approach is therefore similar in spirit to that of [1], except that the expansion is different and the original (ideal) filter is no longer of the form (1.5) but rather a piecewise polynomial function which itself approximates function (1.3) of the TSVD method. Our motivation is to develop a technique that is similar to the conjugate gradient in the sense that it relies solely on matrix-vector products. At the same time, we would like the iteration polynomial of the method, to mimic the effect of the Truncated SVD method. Our methods begins with an ideal high-pass filter which is a piecewise approximation to the function $H(t-a)$. This in turn is approximated by a polynomial of a certain degree on the interval of the eigenvalues of $A$.

The paper is organized as follows. The next section provides a brief overview of regularization techniques. Section 3 describes our approach using polynpmial filters. Section 4 discusses a few convergence results related to the method. Section 5 reports on a few numerical tests. Finally, the paper gives some concluding remarks in Section 6.

**2. Regularization methods.** To grasp the effect of regularization it is helpful to use the Singular Value Decomposition of $A$, see, e.g., [3, 4, 6] for details. We now return to the general situation where $A$ is non-symmetric, possibly rectangular of size $n \times m$ with $n \geq m$. Let $A = U\Sigma V^T$ the Singular Value Decomposition (SVD) of $A$, where $U$ and $V$ are orthonormal bases and $\Sigma$ is a diagonal matrix whose entries are the singular values of $A$

$$\Sigma = \mathrm{diag}\,(\sigma_1, \cdots, \sigma_n)\,, \quad \text{with} \quad \sigma_1 \geq, \cdots, \geq \sigma_n \geq 0.$$

The right-hand-side $b$ is expanded in the $U$-basis as

$$b = \sum_{j=1}^{n} \xi_j u_j, \quad \text{with} \quad \xi_j = u_j^T b, j = 1, \cdots, n.$$

The pseudo-inverse solution of the system can then be written as follows:

$$x^\dagger = A^\dagger\ b\ = \sum_{\sigma_j > 0} \frac{1}{\sigma_j} \xi_j v_j \tag{2.1}$$

in which $A^\dagger$ is the pseudo-inverse of $A$. As can be seen, for small singular values any noise on the component $\xi_j$ will be amplified by $1/\sigma_j$. This will cause noise – which is predominant in the small components – to be amplified to unacceptable levels.

The idea of regularization methods is to introduce filter factors $\phi_j$ in the solution

$$x_\phi = \sum_{\sigma_j > 0} \frac{\phi_j}{\sigma_j} \xi_j v_j\ . \tag{2.2}$$

where $\phi_j$ is usually the value of a filter function $\phi$ at $\sigma_j$. Let $\phi$ and $f$ two functions defined on an interval $[0, g]$ with $\sigma_1 \leq g$ such that

$$\begin{aligned} \phi(0) &= 0, \quad \phi(\sigma_j) = \phi_j, \\ f(0) &= 0, \quad f(t) = \tfrac{\phi(t)}{t}, \quad t \in (0, g]. \end{aligned}$$

Define $\phi(\Sigma)$ and $f(\Sigma)$ to be the diagonal matrices with entries $\phi(\sigma_j)$ and $f(\sigma_j)$ respectively. The filtered solution and the filtered right-hand side are

$$x_\phi = V f(\Sigma) U^T b, \tag{2.3}$$

$$A x_\phi = U \phi(\Sigma) U^T b. \tag{2.4}$$

Note that the residual is given by $b - A x_\phi = U[I - \phi(\Sigma)]U^T b$.

We observe that the regularization dampens each component of the solution by $\phi(\sigma_j)$ before dividing it by $\sigma_j$. It is typical in ill-posed problems to filter out small singular values, so the function $\phi$ is chosen so that

$$\phi(\sigma_j) \simeq 0, \text{ for small } \sigma_j, \qquad \phi(\sigma_j) \simeq 1, \text{ for large } \sigma_j.$$

With regularization, any noise component in the direction $u_j$ is amplified by $\phi(\sigma_j)/\sigma_j$. Thus, the amplification of the noise is bounded, in fact it is even made to tend to zero for small $\sigma_j$'s. Clearly, the regularized solution will differ significantly from the pseudo-inverse solution if there are many small singular values since

$$x^\dagger - x_\phi = \sum_{\sigma_j > 0} \left[ \frac{1 - \phi(\sigma_j)}{\sigma_j} \right] \xi_j y_j.$$

However, it is important to note that the exact solution $x^\dagger$ may be meaningless since it may include very noisy data.

**2.1. Truncated SVD.** The simplest regularization method simply replaces the solution (2.1) by

$$x_\epsilon = \sum_{\sigma_j > \epsilon} \frac{\xi_j}{\sigma_j} v_j \tag{2.5}$$

in which $\epsilon$ is a selected parameter. This is a regularized solution where the filter function $\phi_\epsilon$ is given by (1.3), which is piecewice constant function with a value of zero for $t \leq \epsilon$ and one elsewhere. Regularization based on the SVD approach, requires the Singular Value Decomposition of the matrix $A$, and this is clearly not realistic for large matrices. A number of alternative methods have been developed – the best known of which, Tychonov regularization, is summarized next.

**2.2. Tychonov Regularization.** Tychonov regularization [12, 11, 3] replaces the solution of the original system by that of the following 'regularized' system

$$\min_x \left( \|Ax - b\|^2 + \rho^2 \|x\|^2 \right),$$

which is obtained by solving the system

$$\left( A^T A + \rho^2 I \right) x = A^T b,$$

The solution of the above system is now given by

$$x_\rho = \left( A^T A + \rho^2 I \right)^{-1} A^T b = \phi_\rho(A) A^\dagger b = f_\rho(A) b, \tag{2.6}$$

where $\phi_\rho$ is the Tychonov filter function (1.4).

The above approximation is seldom computed exactly by a direct method. It is instead often obtained by the conjugate gradient algorithm. Since the matrix $A^T A$ is shifted, it is common that the number of steps required by the CG method to converge on such systems is moderate.

A notable drawback of Tychonov regularization is that it tends to produce a solution that is often excessively smooth. In image processing this results in loss of sharpness.

A related method that is quite common in the case of low noise, is simply to use the Conjugate gradient method for solving $A^T Ax = A^T b$ and stop the process prematuraly, i.e., well before convergence of the approximate solution [3].

**2.3. The symmetric case.** When the matrix $A$ is symmetric semi-positive definite, then the $\sigma_i$'s are the eigenvalues $\lambda_i$ of $A$ and the left and right singular vectors are equal to the eigenvectors of $A$, i.e., $v_i = u_i$, for $i = 1, \ldots, n$. In this case, the expressions (2.3) and (2.4) are still valid with $V$ replaced by the matrix of eigenvectors $U$ and $\Sigma$ replaced by the diagonal matrix $\Lambda$ of eigenvalues.

**3. Polynomial Filters.** Consider a symmetric matrix $A$ and a filter function $\phi$, that is suitable for regularizing the original system. Therefore the function $\phi$ satisfies

$$f(0) = 0, \quad \phi(t) = tf(t).$$

Note that the function $\phi$ is defined only for $t \geq 0$ since it will act on singular values (or eigenvalues of semi-positive definite symmetric matrices). We can also consider that the function is defined by symmetry for negative $t$ by $\phi(t) = \phi(-t)$, i.e., that it is an even function.

The function $\phi$ depends on $A$, on $b$, and on knowledge of the problem being modeled. In theory, from the knowledge of the filter function $\phi$, one can easily obtain the regularized solution via the relation (2.3). Thus, the function $f$ yields directly the regularized solution, provided one can easily compute the product of $f(A)$ times a vector. However, for large matrices, the regularized solution $f(A)b$ may be difficult or expensive to compute.

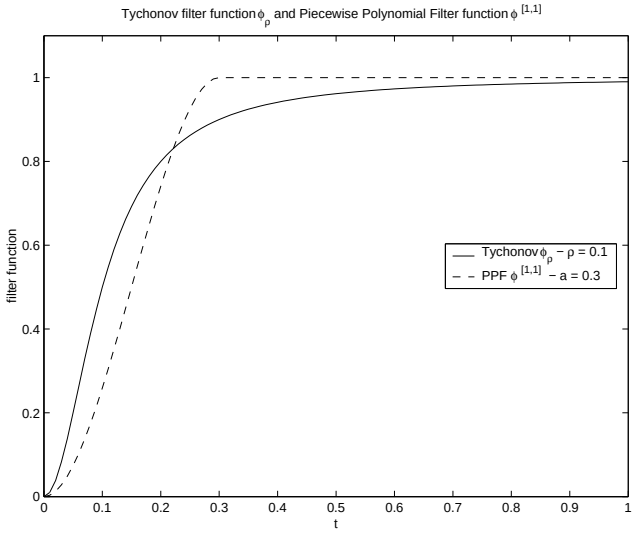



FIG. 3.1. *The Tychonov filter and Piecewise Polynomial filter*

Consider the two examples of the filter functions shown in Figure 3.1. The dashed line is the Tychonov filter with $\rho = 0.1$ and the solid line is the filter piecewise polynomial function defined by

$$\phi^{[1,1]}(t) = \begin{cases} -\frac{2}{a^3}t^3 + \frac{3}{a^2}t^2 & \text{for} \quad t \in [0, a], \\ 1 & \text{for} \quad t \in [a, g], \end{cases} \tag{3.1}$$

with $a = 0.3$.

The problem with these two filter functions, is that they do not lead to an easy evaluation of the regularized solution via (2.3). The Tychonov filtered solution $f_\rho(A)b$ can be computed via an eigenvalue decomposition of $A$ (or an SVD in the least-squares case), but it is only approximated in practice, using an iterative method such as the Conjugate Gradient.

The piecewise polynomial filter $\phi^{[1,1]}$ is even more difficult to approximate. However, the solution proposed in this paper is *to approximate any piecewise polynomial filter by a polynomial in some least-squares sense.*

Define $\mathbb{P}_{k,1}$ the set of polynomials $p$ of degree $k$ such that $p(0) = 0$ and $\mathbb{P}_{k+1,2}$ the set of polynomials $p$ of degree $k+1$ such that $p(0) = p'(0) = 0$ :

$$\mathbb{P}_{k,1} \equiv \langle t, \cdots, t^k \rangle \qquad \text{and} \qquad \mathbb{P}_{k+1,2} \equiv \langle t^2, \cdots, t^{k+1} \rangle$$

The algorithms build a sequence of polynomials $\phi_k$ in $\mathbb{P}_{k+1,2}$ and a sequence of polynomials $f_k$ in $\mathbb{P}_{k,1}$ such that

$$f_k(0) = 0, \quad \phi_k(t) = t f_k(t),$$

and compute a regularized approximation $x_k$ along with an *approximate filtered right-hand side* $Ax_k$ of the form,

$$x_k = f_k(A)b, \qquad Ax_k = \phi_k(A)b.$$

The approximate solution $x_k$ belongs to the Krylov subspace

$$\mathcal{L}_k(A,b) = \mathcal{K}_k(A,Ab) = \langle Ab, \cdots, A^k b\rangle, \tag{3.2}$$

while the approximate filtered right-hand side is in the Krylov subspace

$$\mathcal{R}_k(A,b) = A\mathcal{L}_k(A,b) = \mathcal{K}_k(A,A^2b) = \langle A^2b, \cdots, A^{k+1}b\rangle. \tag{3.3}$$

Note that the space $\mathcal{R}_k$ is not the space of residuals $b - Ax_k$ nor the space of residuals associated with the filtered right-hand side $\phi(A)b$, since neither $b$ nor $\phi(A)b$ belong to $\mathcal{R}_k(A,b)$.

The final requirement for the polynomial $\phi_k$ is that it should approximate the "ideal filter function" $\phi$, in some sense, and converge to it for $k \to \infty$ in some sense. For this we simply define $\phi_k$ as the least-squares approximation to the function $\phi$. Therefore,

$$\phi_k(t) = \sum_{j=1}^{k} \langle \phi, \mathcal{P}_j\rangle \mathcal{P}_j(t), \tag{3.4}$$

where $\{\mathcal{P}_j\}$ is a basis of polynomials that is orthonormal for some $L_2$ inner-product. The $L_2$-inner product will be selected essentially to make the computation tractable without resorting to numerical integration. The next section desribes the overall procedure in detail.

**3.1. Use of an orthogonal basis of polynomials.** For the sake of clarity, we delay the discussion of the choice of the $L_2$ inner-product until later in Section 3.5. Assume therefore that the $L_2$ inner-product for the above least-squares approximation is defined and that computing such $L_2$-inner products of polynomials is inexpensively accomplished. It is helpful, for computational purposes, to use an orthonormal basis of polynomials associated with this inner product. This is traditionally done with the well-known Stiejes procedure, which utilizes a 3-term recurrence [2]. Starting with a certain polynomial $\mathcal{S}_0$ the sequence is built as follows.

ALGORITHM 3.1. **Stieljes**

1. $\mathcal{P}_0 \equiv 0$,
2. $\beta_1 = \|\mathcal{S}_0\|_{\langle\ \ \rangle}$,
3. $\mathcal{P}_1(t) = \frac{1}{\beta_1}\mathcal{S}_0(t)$,
4. *For* $j = 2, \ldots, m$ *Do*
5. $\quad \alpha_j = \langle t\ \mathcal{P}_j, \mathcal{P}_j\rangle$,
6. $\quad \mathcal{S}_j(t) = t\ \mathcal{P}_j(t) - \alpha_j \mathcal{P}_j(t) - \beta_j \mathcal{P}_{j-1}(t)$,
7. $\quad \beta_{j+1} = \|\mathcal{S}_j\|_{\langle\ \ \rangle}$,
8. $\quad \mathcal{P}_{j+1}(t) = \frac{1}{\beta_{j+1}}\mathcal{S}_j(t)$.
9. *EndDo*

Note that for convenience the first polynomial in the sequence is $\mathcal{P}_1$ (with $\mathcal{P}_0 \equiv 0$) instead of being $\mathcal{P}_0$ (with $\mathcal{P}_{-1} \equiv 0$) as is common practice. The polynomials satisfy the three-term recurrence

$$\mathcal{P}_{j+1}(t) = \frac{1}{\beta_{j+1}} \left( t\, \mathcal{P}_j(t) - \alpha_j \mathcal{P}_j(t) - \beta_j \mathcal{P}_{j-1}(t) \right) \quad j = 1, \ldots, m \tag{3.5}$$

The choice of $\mathcal{S}_0$ will depend on the space for which we are building a basis. Here the filter functions $\phi_k$ are in the space $\mathbb{P}_{k+1,2}$ so that $\mathcal{S}_0(t) = t^2$. The above procedure requires only to be able to compute (1) inner products of polynomials, (2) the product $t \times p$ for a given polynomial $p$ and (3) linear combinations of polynomials. The details on how to perform these operations will be discussed in Sections 3.5 and Section 3.6.

**3.2. The polynomial to solution space isomorphism.** Let $\mu$ be the maximum dimension of the subspaces $\mathcal{L}_k$ and $\mathcal{R}_k$ for $k \geq 1$. A strong relationship can be established between the filtered space $\mathcal{R}_k$ and the poynomial space $\mathbb{P}_{k+1,2}$, which will be quite helpful. This relationship is the mapping

$$\phi_k \in \mathbb{P}_{k+1,2} \;\rightarrow\; \phi_k(A)b \;\in \mathcal{R}_k(A, b) \tag{3.6}$$

When $k \leq \mu \leq n$, this mapping is a bijection.

What is interesting about this mapping is that it allows us to provide the Krylov subspace $\mathcal{R}_k(A, b)$ with a dot product, which is canonically derived from the polynomial space. As a result of this we obtain an isomorphism between the two spaces which allows us, with few exceptions, to utilize most of the well-known CG-type algorithms for computing the approximate solution $x_k$.

We define the 'Lanczos' sequence $v_j$ with respect to this inner product by

$$v_j = \mathcal{P}_j(A)b$$

With this isomorphism, a number of standard operations in the filtered space have their immediate analogues in the polynomial space.

| Filtered space | Polynomial space |
|---|---|
| Addition of vectors | Addition of polynomials |
| Scaling of vectors | scaling of polynomials |
| Inner products | inner products |
| Lanczos Algorithm | Stieljes procedure |
| 3-term recurrence | 3-term recurrence |
| $v_j$ 'Lanczos' basis vector | orthogonal polynomial $\mathcal{P}_j$ |
| Filtered sequence $Ax_k$ | polynomial sequence $\phi_k$ |

In particular, the vectors $v_j$ verify the 3-term recurrence

$$v_{j+1} = \frac{1}{\beta_{j+1}} \left( Av_j - \alpha_j v_j - \beta_j v_{j-1} \right), \quad \forall j \geq 1, v_1 = \frac{1}{\beta_1} A^2 b, v_0 = 0. \tag{3.7}$$

which is simply the Lanczos procedure, with the usual Euclidean inner-product replaced by the inner-product defined from the polynomial space.

**3.3. The algorithm.** We can now write directly the $k$-th solution vector $x_k$ in terms of a sequence of vectors that are related to the sequence of orthogonal polynomials. A first observation, using (3.4) is that

$$\phi_k(t) = \sum_{j=1}^{k} \langle \phi, \mathcal{P}_j \rangle \mathcal{P}_j(t) \qquad \rightarrow \qquad f_k(t) = \sum_{j=1}^{k} \langle \phi, \mathcal{P}_j \rangle \frac{\mathcal{P}_j(t)}{t}$$

Let $\gamma_j = \langle \phi, \mathcal{P}_j \rangle$ and $\mathcal{Q}_j(t) = \mathcal{P}_j(t)/t$. The sequence of vectors $w_j = \mathcal{Q}_j(A)b$ is in the solution space $\mathcal{L}_k(A, b)$. From this follows the formal expansion

$$x_k = \sum_{j=1}^{k} \gamma_j w_j. \tag{3.8}$$

Because of the the way the algorithm is started, the sequence of vectors $w_j$ is easily computable. Indeed,

$$w_1 = \frac{1}{\beta_1} Ab, \quad w_0 = 0 \ .$$

Thereafter, it is sufficient to divide the recurrence relation (3.5) by $t$ to obtain immediatly the following recurrence for the sequence of vectors $w_j$:

$$w_{j+1} = \frac{1}{\beta_{j+1}} \left( v_j - \alpha_j w_j - \beta_j w_{j-1} \right) \quad \forall j \geq 1. \tag{3.9}$$

The algorithm is now easy to describe.

ALGORITHM 3.2. **PPF : Regularization by Piecewise Polynomial Filter**

0. **Input:** *$\phi$ piecewise polynomial on $[0, g]$*
1. $\beta_1 = \langle t^2, t^2 \rangle^{\frac{1}{2}}$
2. $\mathcal{P}_1(t) = \frac{1}{\beta_1} t^2$
3. $\gamma_1 = \langle \phi, \mathcal{P}_1 \rangle$
4. $v_1 = \frac{1}{\beta_1} A^2 b$, $v_0 = 0$
5. $w_1 = \frac{1}{\beta_1} Ab$, $w_0 = 0$
6. $x_1 = \gamma_1 w_1$, $b_1 = \gamma_1 v_1$
7. **For** $k = 1, \ldots$ **Do:**
8. *Compute $t\, \mathcal{P}_k(t)$*
9. $\alpha_k = \langle t\, \mathcal{P}_k, \mathcal{P}_k \rangle$
10. $\mathcal{S}_k(t) = t\, \mathcal{P}_k(t) - \alpha_k \mathcal{P}_k(t) - \beta_k \mathcal{P}_{k-1}(t)$
11. $s_k = A v_k - \alpha_k v_k - \beta_k v_{k-1}$
12. $\beta_{k+1} = \langle \mathcal{S}_k, \mathcal{S}_k \rangle^{\frac{1}{2}}$
13. $\mathcal{P}_{k+1}(t) = \frac{1}{\beta_{k+1}} \mathcal{S}_k(t)$
14. $v_{k+1} = \frac{1}{\beta_{k+1}} s_k$
15. $\gamma_{k+1} = \langle \mathcal{P}_{k+1}, \phi \rangle$

16. $w_{k+1} = \frac{1}{\beta_{k+1}} (v_k - \alpha_k w_k - \beta_k w_{k-1})$
17. $x_{k+1} = x_k + \gamma_{k+1} w_{k+1}$
18. **EndDo**

Note that it is possible to compute the sequence of related approximations $b_k$ to the filtered right-hand side, by adding the line

$$b_{k+1} = b_k + \gamma_{k+1} v_{k+1}$$

immediatly after Line 17. Next, we shed some light on the choice of the filter function $\phi$ upon which the whole procedure is based.

**3.4. General bridge functions.** In defining the ideal filter polynomial $\phi(t)$, we invoked a piecewise polynomial which has value one for $t$ in the interval $[a, g]$ and which moves smoothly from 0 to 1 as $t$ moves from 0 to $a$. The first part of this peciewise function can be viewed as a "bridge" which joins continously the constant function zero for $t \leq 0$ to the constant function one for $t \geq a$. We impose smoothness conditions on this function – for example by requiring that a maximum number of derivatives at zero and $a$ be equal to zero. The simplest of these functions is the filter function $\phi^{[1,1]}$ defined by (3.1). It has value zero at zero, and one at $a$ and its derivatives at zero and $a$ are zero. This function and more general ones like it, can be systematically generated by using Hermite interpolation. It is useful to define bridge functions of arbitrary degree of smoothness.

In order to generalize the function in (3.1), we can require the following conditions

$$\begin{array}{lcllcll} \phi(0) & = & 0 & & \phi(a) & = & 1 \\ \phi^{(i)}(0) & = & 0 & \text{for } i = 1, \ldots, m & \phi^{(i)}(a) & = & 0 & \text{for } i = 1, \ldots, p \end{array}$$

There are $m+p+2$ conditions altogether and therefore a unique polynomial $\phi^{[m,p]}$ of degree $m + p + 1$ can be found which satisfies them. Such a polynomial can be easily determined by the usual finite difference tables in the Hermite sense.

There are interesting questions related to the quality of the filter functions obtained as a function of the two values $m$ and $p$. To find a closed form for the polynomials $\phi^{[m,p]}$ it is useful to change variables in order to exploit symmetry. We translate everything for the variable in the interval $[-1, 1]$, and shift down the function by $1/2$. Then the above conditions become

$$\begin{array}{lcllcll} \eta(-1) & = & -1/2 & & \eta(+1) & = & 1/2 \\ \eta^{(i)}(-1) & = & 0 & \text{for } i = 1, \ldots, m & \eta^{(i)}(+1) & = & 0 & \text{for } i = 1, \ldots, p \end{array}$$

The derivative function $\eta'$ is sought in the form

$$\eta'(t) = c\ (1-t)^p (1+t)^m$$

It is easily seen that the function defined by

$$\eta(t) = -\frac{1}{2} + \frac{\int_{-1}^{t} (1-s)^p (1+s)^m\ ds}{\int_{-1}^{1} (1-s)^p (1+s)^m\ ds} \tag{3.10}$$

satisfies all the required conditions stated above. The derivative of $\eta$ is

$$\eta'(t) = c\ (1-t)^p(1+t)^m$$

where the constant $c$ is the integral in the denominator of (3.10). The second derivative is given by

$$\begin{aligned}\eta''(t) &= c\left[m(1-t)^p(1+t)^{m-1} - p(1-t)^{p-1}(1+t)^m\right] \\ &= c(1-t)^{p-1}(1+t)^{m-1}\left[m(1-t) - p(1+t)\right]\end{aligned}$$

So there is an inflexion point at

$$t = \frac{m-p}{m+p}.$$

Translating this into the original interval $[0, a]$ gives,

$$t_{infl} = \frac{a}{2}\ \left[1 + \frac{m-p}{m+p}\right] = \frac{m \times a}{m+p} = \frac{a}{1+p/m}$$

At the right of the inflexion point, the filter function increases rapidly toward one. To its left it decreases rapidly toward zero. This can be exploited in deciding of a zero-out zone where noise (as well as solution components) must be eliminated and a grey area, where noise is unimportant and need not be eliminated completely.

In what follows, we show some examples of bridge functions for the cases when $m = p$. When $m = p = 1$ we find that

$$\eta(t) = \frac{3}{4}\left(t - \frac{t^3}{3}\right) = \frac{3t}{4} - \frac{t^3}{4}$$

which, after shifting back up by $1/2$ and translation back to the interval $[0, a]$ yields the 3rd degree bridge function

$$\phi^{[1,1]}(t) = \frac{1}{2} + \frac{3}{4}\left(2\frac{t}{a} - 1\right) - \frac{1}{4}\left(2\frac{t}{a} - 1\right)^3$$

After a few simplifications, this yields the same function $3(t/a)^2 - 2(t/a)^3$ used in (3.1). Similarly, for $m = p = 2$, we get

$$\eta(t) = \left(t - 2\frac{t^3}{3} + \frac{t^5}{5}\right)\ \times\ \frac{15}{16}$$

going back again to the original variables and reshifting yileds,

$$\phi^{[2,2]}(t) = \frac{1}{2} + \frac{15}{16}\ \left(2\frac{t}{a} - 1\right) - \frac{5}{8}\ \left(2\frac{t}{a} - 1\right)^3 + \frac{3}{16}\left(2\frac{t}{a} - 1\right)^5$$

The last bridge function we show is the one obtained for $m = p = 3$:

$$\phi^{[3,3]}(t) = \frac{1}{2} + \frac{35}{32}\left(2\frac{t}{a} - 1\right) - \frac{35}{32}\left(2\frac{t}{a} - 1\right)^3 + \frac{21}{32}\left(2\frac{t}{a} - 1\right)^5 - \frac{5}{32}\left(2\frac{t}{a} - 1\right)^7$$

The three bridge functions $\phi^{[1,1]}$, $\phi^{[3,1]}$ and $\phi^{[1,3]}$ are illustrated in Figure 3.2.

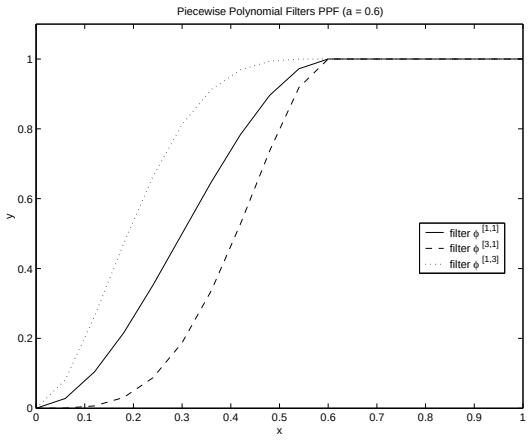



FIG. 3.2. *The three bridge filter functions* $\phi^{[1,1]}$, $\phi^{[3,1]}$, $\phi^{[1,3]}$

**3.5. Inner product.** We consider the situation described above when the original filter function $\phi$ to be approximated is a piecewise polynomial function in an interval $[0, g]$ containing $[\lambda_n, \lambda_1]$. The interval $[0, g]$ is subdivided into $L$ sub-intervals such that

$$[0, g] = \bigcup_{l=1}^{L} [a_{l-1}, a_l] \qquad \text{with} \qquad a_0 = 0, \qquad a_L = g > \lambda_1$$

Note that, if the matrix is not singular, the leftmost bound of zero can be changed to an arbitrary positive number but this is not considered here. A reasonable value for $g$, and an inexpensive one to compute, can be obtained by using Gershgorin's theorem. In the case $L = 2$, we denote by $a$ the right bound $a_1$ of the first interval.

The Stieljes procedure as well as the least-squares procedure, requires the computation of inner products for calculating the scalars $\alpha_j$ and $\beta_{j+1}$, and expansion coefficients $\gamma_j$. The inner product defined for computing the least-squares polynomial is selected to allow an easy calculation of these scalars. It is based on an approach used in [9, 10] for solving a similar problem related to indefinite linear systems of equations. The idea of using a step function (Heaviside function) as an ideal filter and then approximate it by polynomials in the least-squares sense was also used in a very different context in Chemistry, see e.g., [8, 7], and references therein.

The inner product utilizes the same subdivision of the interval $[0, g]$ as that of the piecewise polynomial function. On each subinterval $[a_{l-1}, a_l]$ of the subdivision we define the inner-product $\langle \psi_1, \psi_2 \rangle_{a_{l-1}, a_l}$ by

$$\langle \psi_1, \psi_2 \rangle_{a_{l-1}, a_l} = \int_{a_{l-1}}^{a_l} \frac{\psi_1(t)\psi_2(t)}{\sqrt{(t - a_{l-1})(a_l - t)}}\, dt.$$

Then the inner product on the interval $[0, g]$ is defined as the weighted sum of the inner products on the smaller intervals:

$$\langle \psi_1, \psi_2 \rangle = \sum_{l=1}^{L} \rho_l \ \langle \psi_1, \psi_2 \rangle_{a_{l-1}, a_l} \tag{3.11}$$

For the particular case when $L = 2$ this can be rescaled as:

$$\langle \psi_1, \psi_2 \rangle = \int_0^a \frac{\psi_1(t)\psi_2(t)}{\sqrt{t(a - t)}}\, dt + \rho \int_a^g \frac{\psi_1(t)\psi_2(t)}{\sqrt{(t - a)(g - t)}}\, dt. \tag{3.12}$$

In [9] a basis of Chebyshev polynomials on each of the two intervals was used, in order to avoid numerical integration. Let $\varsigma^{(l)}$ the mapping which transforms the interval $[a_{l-1}, a_l]$ into $[-1, 1]$:

$$\varsigma^{(l)}(t) = \frac{2}{a_l - a_{l-1}} t - \frac{a_l + a_{l-1}}{a_l - a_{l-1}}$$

Denote by $C_i$ the $i$-th Chebyshev polynomial on $[-1, 1]$ and define

$$C_i^{(l)}(t) = C_i \left( \varsigma^{(l)}(t) \right) \quad i \geq 0.$$

Then, clearly, $C_0^{(l)}(t) = 1$, $C_1^{(l)}(t) = \varsigma^{(l)}(t)$ and for $i \geq 1$ we have the recurrence relation

$$\begin{aligned} C_{i+1}^{(l)}(t) &= 2\varsigma^{(l)}(t) C_i \left( \varsigma^{(l)}(t) \right) - C_{i-1} \left( \varsigma^{(l)}(t) \right), \\ &= \frac{4}{a_l - a_{l-1}} t\ C_i^{(l)}(t) - 2\frac{a_l + a_{l-1}}{a_l - a_{l-1}} C_i^{(l)}(t) - C_{i-1}^{(l)}(t), \end{aligned}$$

The following relation will be useful,

$$t\ C_i^{(l)}(t) = \frac{a_l - a_{l-1}}{4} C_{i+1}^{(l)}(t) + \frac{a_l + a_{l-1}}{2} C_i^{(l)}(t) + \frac{a_l - a_{l-1}}{4} C_{i-1}^{(l)}(t) \quad i \geq 1 \tag{3.13}$$

$$t\ C_0^{(l)}(t) = \frac{a_l - a_{l-1}}{2} C_1^{(l)}(t) + \frac{a_l + a_{l-1}}{2} C_0^{(l)}(t) \tag{3.14}$$

Recall that on each interval these scaled and shifted Chebyshev polynmials $\left( C_k^{(l)} \right)_{k \in \mathbb{N}}$ constitute an orthogonal basis since,

$$\langle C_i^{(l)}, C_j^{(l)} \rangle_{a_{l-1}, a_l} = \begin{cases} 0 & \text{if } i \neq j, \\ \pi & \text{if } i = j = 0, \\ \frac{\pi}{2} & \text{if } i = j \neq 0. \end{cases}$$

**3.6. Computation of Stieljes coefficients.** As was mentioned earlier, calculations in the Stieljes procedure will be faciliated by the use of a redundant representation of the polynomials in the Chebyshev bases on each subinterval. This technique follows closely a procedure defined in [9]. Specifically, the $j$-th polynomial $\mathcal{P}_j$ is represented in the Chebyshev basis on the $l$-th interval, for $1 \leq l \leq L$, as :

$$\mathcal{P}_j(t) = \sum_{i=0}^{j} \mu_{i,j}^{(l)} C_i^{(l)}(t) \tag{3.15}$$

As a convention we define $\mu_{j+1,j}^{(l)} \equiv 0$ in what follows. We also set

$$t\ \mathcal{P}_j(t) = \sum_{i=0}^{j+1} \sigma_{i,j}^{(l)} C_i^{(l)}(t).$$

The $\sigma_{i,j}$'s can be obtained from the $\mu_{i,j}$'s with the help of the recurrence (3.13). Indeed,

$$\begin{aligned} t\ \mathcal{P}_j(t) &= t \sum_{i=0}^{j} \mu_{i,j}^{(l)} C_i^{(l)}(t) = \sum_{i=0}^{j} t\mu_{i,j}^{(l)} C_i^{(l)}(t) \\ &= \sum_{i=0}^{j} \mu_{i,j}^{(l)} \left( \frac{a_l - a_{l-1}}{4} C_{i+1}^{(l)}(t) + \frac{a_l + a_{l-1}}{2} C_i^{(l)}(t) + \frac{a_l - a_{l-1}}{4} C_{i-1}^{(l)}(t) \right) \\ &= \sum_{i=0}^{j+1} \left( \frac{a_l - a_{l-1}}{4} (\mu_{i+1,j}^{(l)} + \mu_{i-1,j}^{(l)}) + \frac{a_l + a_{l-1}}{2} \mu_{i,j}^{(l)} \right) C_i^{(l)}(t). \end{aligned}$$

Hence,

$$\sigma_{i,j}^{(l)} = \frac{a_l - a_{l-1}}{4} \left( \mu_{i+1,j}^{(l)} + \mu_{i-1,j}^{(l)} \right) + \frac{a_l + a_{l-1}}{2} \mu_{i,j}^{(l)}. \tag{3.16}$$

Now, once the components of $\mathcal{P}_j$ and $t\ \mathcal{P}_i$ are known on all the bases $\left( C_j^{(l)} \right)$, it is easy to compute the partial inner products :

$$\begin{aligned} \langle t\ \mathcal{P}_j, \mathcal{P}_j \rangle_{a_{l-1}, a_l} &= \left\langle \sum_i \sigma_{i,j}^{(l)} C_i, \sum_i \mu_{i,j}^{(l)} C_i \right\rangle_{a_{l-1}, a_l}, \\ &= \pi \sigma_{0,j}^{(l)} \mu_{0,j}^{(l)} + \frac{\pi}{2} \sum_{i=1}^{j+1} \sigma_{i,j}^{(l)} \mu_{i,j}^{(l)}. \end{aligned}$$

from which we can extract the value of $\alpha_j$ :

$$\alpha_j = \langle t\ \mathcal{P}_j, \mathcal{P}_j \rangle = \sum_{l=1}^{L} \rho_l \langle t\ \mathcal{P}_j, \mathcal{P}_j \rangle_{a_{l-1}, a_l}$$

so that

$$\alpha_j = \pi \sum_{l=1}^{L} \rho_l \left( \sigma_{0,j}^{(l)} \mu_{0,j}^{(l)} + \frac{1}{2} \sum_{i=1}^{j+1} \sigma_{i,j}^{(l)} \mu_{i,j}^{(l)} \right), \tag{3.17}$$

If we now define

$$\mathcal{S}_j = \sum_{i=0}^{j+1} \eta_{i,j}^{(l)} C_i^{(l)}.$$

then because $\mathcal{S}_j(t) = t\ \mathcal{P}_j(t) - \alpha_j \mathcal{P}_j(t) - \beta_j \mathcal{P}_{j-1}(t)$ we readily obtain the relation:

$$\eta_{i,j}^{(l)} = \sigma_{i,j}^{(l)} - \alpha_j \mu_{i,j}^{(l)} - \beta_j \mu_{i,j-1}^{(l)}, \tag{3.18}$$

This enables us to compute $\beta_{j+1} = \langle \mathcal{S}_j, \mathcal{S}_j \rangle^{\frac{1}{2}}$ since

$$\beta_{j+1}^2 = \pi \sum_{l=1}^{L} \rho_l \left( \eta_{0,j}^2 + \frac{1}{2} \sum_{i=1}^{j+1} \left( \eta_{i,j}^{(l)} \right)^2 \right) \tag{3.19}$$

and then we have

$$\mu_{i,j+1}^{(l)} = \frac{1}{\beta_{j+1}} \eta_{i,j}^{(l)}. \tag{3.20}$$

In summary, equation (3.16) is used to compute $t\mathcal{P}_j$ and the equations (3.17-3.20) are used to compute the scalars $\alpha_j$ and $\beta_j$. We also have to compute the inner products $\gamma_j$. This is easily done because the filter function $\phi$ is piecewise polynomial and is readily expanded in the Chebyshev basis on each subinterval.

**4. Convergence Analysis.** The lemma and the proposition which follow will provide an upper bound on the $\| \ \|_{\langle\ ,\ \rangle}$, from the infinity norm. Only the case case $L = 2$ is considered. First notice that

$$\|\phi - \phi_k\| = \mathrm{d}\left(\phi, \mathbb{P}_{k+1,2}\right)_{\langle\ ,\ \rangle}.$$

where $d(f, \mathbb{S})$ is the $L_2$ distance between a function $f$ and the function space $\mathbb{S}$. This is simply because $\phi_k$ is, by definition, the orthogonal projection $\langle\ ,\ \rangle$ of $\phi$ onto $\mathbb{P}_{k+1,2}$. We now can state the following

LEMMA 4.1. *For an inner product $\langle\ ,\ \rangle$ defined by (3.12) the following upper bound holds,*

$$\|\psi\|_{\langle\ ,\ \rangle}^2 \leq (1+\rho)\pi \|\psi\|_\infty^2. \tag{4.1}$$

**Proof.** We have indeed:

$$\begin{aligned}\|\psi\|^2_{\langle\ ,\ \rangle} &= \int_0^a \frac{\psi^2(t)}{\sqrt{t(a-t)}}\,dt + \rho\int_a^g \frac{\psi^2(t)}{\sqrt{(t-a)(b-t)}}\,dt \\ &\le \|\psi\|^2_\infty \left(\int_0^a \frac{1}{\sqrt{t(a-t)}}\,dt + \rho\int_a^g \frac{1}{\sqrt{(t-a)(b-t)}}\,dt\right) \\ &\le (1+\rho)\pi\|\psi\|^2_\infty.\end{aligned}$$

■

PROPOSITION 4.2. $\|\phi - \phi_k\|_{\langle\ ,\ \rangle} \le \sqrt{(1+\rho)\pi}\ \mathrm{d}(\phi, \mathbb{P}_{k+1,2})_\infty$.

**Proof.** The result follows from :

$$\begin{aligned}\|\phi - \phi_k\|_{\langle\ ,\ \rangle} &= \min_{\mathcal{P}\in\mathbb{P}_{k+1,2}} \|\phi - \mathcal{P}\|_{\langle\ ,\ \rangle} \\ &\le \sqrt{(1+\rho)\pi}\min_{\mathcal{P}\in\mathbb{P}_{k+1,2}} \|\phi - \mathcal{P}\|_\infty, \quad \text{from Lemma 4.1} \\ &= \sqrt{(1+\rho)\pi}\ \mathrm{d}(\phi, \mathbb{P}_{k+1,2})_\infty.\end{aligned}$$

■

In order to get estimates of this distance we now introduce Berstein polynomials associated with a function $f$. This is a sequence of polynomials which converges uniformly toward the function $f$. This will then yield an upper bound for $\mathrm{d}(\phi, \mathbb{P}_{k+1,2})_\infty$.

For $f$ defined on $[0,1]$, the $n$-th Berstein polynomial is:

$$B_n(f)(t) = \sum_{k=0}^n f\left(\frac{k}{n}\right)\begin{pmatrix} n \\ k\end{pmatrix} t^k(1-t)^{n-k}.$$

We have $B_n(f)(0) = f(0) = 0$ and $B_n'(f)(0) = nf\left(\frac{1}{n}\right)$ which converges to $f'(0) = 0$.

THEOREM 4.3. *Let $f$ be $\nu$-lipschitz on $[0,1]$ and let $M$ an upper bound of $f$, then*

$$\|f - B_n(f)\|_\infty \le \frac{3}{2}M^{\frac{1}{3}}\nu^{\frac{2}{3}}\ \frac{1}{\sqrt[3]{n}}.$$

**Proof.** We have

$$\begin{aligned}f(x) - B_n(f,x) &= \sum_{k=0}^n \left(f(x) - f\left(\frac{k}{n}\right)\right)\begin{pmatrix} n \\ k\end{pmatrix} x^k(1-x)^{n-k} \\ &= \sum_{\left|\frac{k}{n}-x\right|<\delta} \left(f(x) - f\left(\frac{k}{n}\right)\right)\begin{pmatrix} n \\ k\end{pmatrix} x^k(1-x)^{n-k}\end{aligned}$$

$$+ \sum_{\left|\frac{k}{n}-x\right|>\delta} \left(f(x) - f\left(\frac{k}{n}\right)\right) \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k}$$

Since $f$ is continuous on $[0,1]$ which is compact, it is bounded by $M$. In addition, we have $|f(y) - f(x)| < \nu\delta$ for any $\delta > 0$ and $\forall x, y$ such that $|y - x| < \delta$. Therefore,

$$\begin{aligned}
|f(x) - B_n(f,x)| &\leq \sum_{\left|\frac{k}{n}-x\right|<\delta} \left|f(x) - f\left(\frac{k}{n}\right)\right| \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k} \\
&\quad + \sum_{\left|\frac{k}{n}-x\right|>\delta} \left|f(x) - f\left(\frac{k}{n}\right)\right| \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k} \\
&\leq \nu\delta \sum_{\left|\frac{k}{n}-x\right|<\delta} \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k} + 2M \sum_{\left|\frac{k}{n}-x\right|>\delta} \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k} \\
&\leq \nu\delta + \frac{M}{2n\delta^2},
\end{aligned}$$

since $\sum_{\left|\frac{k}{n}-x\right|>\delta} \left( \begin{array}{c} n \\ k \end{array} \right) x^k (1-x)^{n-k} \leq \frac{1}{4n\delta^2}$. This is true for any $\delta > 0$, it is true in particular for the minimum of the function $\delta \mapsto \nu\delta + \frac{M}{2n\delta^2}$, whose value is $\frac{3}{2}\nu^{\frac{2}{3}} M^{\frac{1}{3}} \frac{1}{\sqrt[3]{n}}$. ■

COROLLARY 4.4. *Let $\phi \in \mathcal{C}^1([0,g])$. Then $\|\phi - B_k(\phi)\|_\infty \leq \frac{3}{2} g^{\frac{2}{3}} \|\phi\|^{\frac{1}{3}} \|\phi'\|^{\frac{2}{3}} . \frac{1}{\sqrt[3]{k}}$.*

**Proof.** Set $f(y) = \phi(gy)$ ; $f$ is defined on $[0,1]$ and is $g\|\phi'\|$-Lipschitz. ■
We set in what follows $\bar{M} = \frac{3}{2} g^{\frac{2}{3}} \|\phi\|^{\frac{1}{3}} \|\phi'\|^{\frac{2}{3}}$, in order to have $\|\phi - \mathrm{B}_k(\phi)\|_\infty \leq \bar{M} \frac{1}{\sqrt[3]{k}}$. Theorem 4.3 does not allow to obtain a result directly because $B_k(\phi) \notin \mathbb{P}_{k+1,2}$. Indeed, $B'_k(\phi)(0) \neq 0$. Therefore, we need to apply the theorem to $\phi - B_k(\phi) - tB'_k(\phi)(0)$, which is in $\mathbb{P}_{k+1,2}$. The lemma 4.5 will yield an upper bound of $\|tB'_k(\phi)(0)\|_\infty$. Then the following theorem (Theorem 4.6) will provide an expression for the convergence rate of the filter.

LEMMA 4.5. *Let $\phi \in \mathcal{C}^1([0,g])$. Assume that there is an $h > 0$, such that $\phi$ is twice differentiable on $[0,h]$ and that $\phi''$ is bounded on $[0,h]$. Then $k\phi\left(\frac{1}{k}\right) \leq \frac{1}{k} \max_{t\in[0,h]} |\phi''(t)|$.*
**Proof.** From the Taylor-Lagrange equality, we have

$$\left|\phi\left(\frac{1}{k}\right) - \phi(0) - \frac{1}{k}\phi'(0)\right| \leq \frac{1}{k^2} \max_{t\in[0,h]} |\phi''(t)|.$$

Hence $k\phi\left(\frac{1}{k}\right) \leq \frac{1}{k} \max_{t\in[0,h]} |\phi''(t)|$. ■

THEOREM 4.6. *Let $\phi \in \mathcal{C}^1([0,g])$. Assume that there is an $h > 0$, such that $\phi$ is twice differentiable on $[0,h]$ and that $\phi''$ is bounded on $[0,h]$. Then $\|\phi - \phi_k\|_{\langle,\rangle} \in O\left(\frac{1}{\sqrt[3]{k}}\right)$.*
**Proof.**

$$\|\phi - \phi_k\|_{\langle,\rangle} \leq \|\phi - (B_k(\phi) - tB'_k(\phi)(0))\|_{\langle,\rangle}$$

$$
\begin{aligned}
&\leq \sqrt{(1+\rho)\pi}\|\phi - (B_k(\phi) - tB_k'(\phi)(0))\,\|_\infty \\
&\leq \sqrt{(1+\rho)\pi}\,(\|\phi - B_k(\phi)\|_\infty + \|tB_k'(\phi)(0)\|_\infty) \\
&\leq \sqrt{(1+\rho)\pi}\left(\frac{\bar{M}}{\sqrt[3]{k}} + gk\phi\left(\frac{1}{k}\right)\right) \\
&\leq \sqrt{(1+\rho)\pi}\left(\frac{\bar{M}}{\sqrt[3]{k}} + \frac{g\|\phi''\|}{k}\right).
\end{aligned}
$$

This implies that $\|\phi - \phi_k\|_{\langle,\rangle} \in O\left(\frac{1}{\sqrt[3]{k}}\right)$. ■

The assumptions of the theorem 4.6 are clearly always satisfied when $\phi$ is defined to be piecewise polynomial and in $\mathcal{C}^1([0,g])$.

## 5. Numerical Experiments.

**5.1. Description of tests.** Algorithm 3.2 was implemented in Matlab on a SPARC workstation. We have experimented the algorithm on image restoration problems, though it can be applied to other ill-conditioned problems as well. For the numerical tests, we generated various problems by blurring an image, and adding noise to the result. Here the blurring matrix is the matrix generated by the function *blur(N,band,smooth)* from the *Regularization Tools package [5]*. The function *blur(N,band,smooth)* has three parameters : the size $N = \sqrt{n}$, the bandwidth *band*, and the amount of smoothing controlled by *smooth*.

The resulting matrix is symmetric with all its eigenvalues between 0 and 1. We have used $L = 2$ for all our filter functions with $g = 1$ as the upper bound for the interval on which the filter $\phi$ is defined. The inner interval bound $a$ is varied between 0 and 1.

The original image is of size N by N and each pixel is coded on grey levels. Therefore, the solution is the columnwise stacked vector $x$ of each coded pixel, while the blurred image is the vector $\overline{b} = Ax$.

Noise is added to each component after blurring. The noise is Gaussian of zero mean and standard deviation $\sigma$. The resulting image is coded by the vector $b = \overline{b} + e$, where $e$ is the vector of noise.

The images are deblurred by three methods : Truncated Singular Value decomposition noted TSVD, Tychonov coupled with Conjugate Gradient noted TCG, the Piecewise Polynomial Filter proposed in this paper. The bridge functions $\phi^{[m,p]}$ defined in section 3.4 are used and the method is noted either PPF(m,p) or PPF for the default values $m = p = 1$.

**5.2. Convergence of the polynomials.** We first examine the convergence of the polynomials $\phi_k$ and $f_k$ toward $\phi$ and $f$. Figure 5.1 shows the convergence of $\gamma_k$. On the left is the history of $\gamma_k$ for three different values of $a$ with $\phi = \phi^{[1,1]}$. On the right is the same history for three different pairs of values $(m,p)$ of the filter function $\phi^{[m,p]}$ with a fixed value $a = 0.6$. Observe that the coefficients decay very rapidly toward zero. The rate of convergence increases with $a$. It also increases with $m$ and the coefficients become smoother when $p$ increases. Figure 5.2 shows the piecewise polynomial filter $\phi^{[1,1]}$ with $a = 0.6$ and the polynomial filters $\phi_k$ of several degrees $k = 2, 4, 8$.

FIG. 5.1. *Convergence of factors* $\gamma_k$

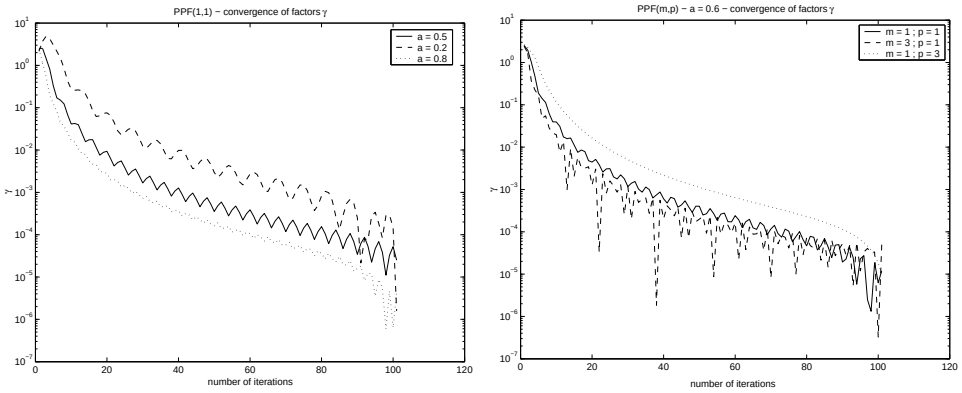
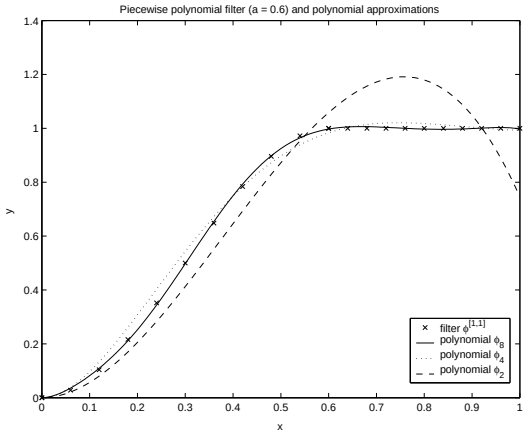
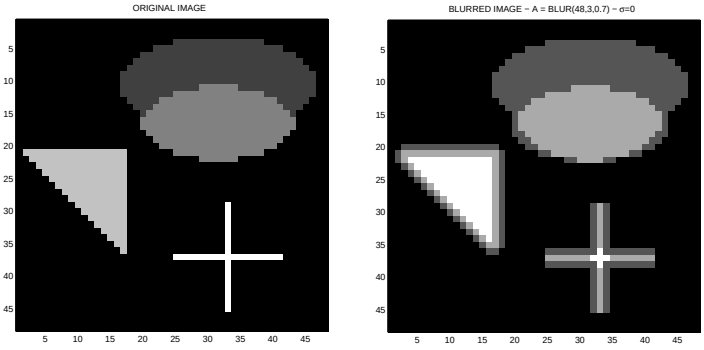



FIG. 5.2. *PPF* $\phi^{[1,1]}$ *and polynomial approximations* $\phi_k$



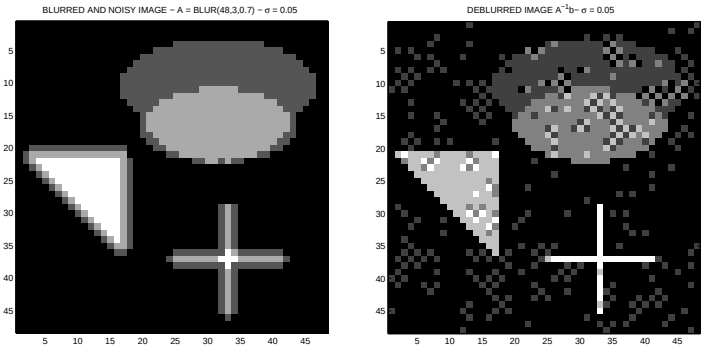
**5.3. Examples 1-4 : an image of medium size.** In this test, the blurring matrix is $A = blur(48, 3, 0.7)$, where *band* and *smooth* are the default values, so the matrix order is $n = 2304$. The original image $x$ is the test image, called MIRE in the following, given by the function *blur*. Here the pixels are coded by integers between 0 and 4. So, for plotting the images, we round all pixels to integers and force them to be in the interval $[0, 4]$. On the other hand, errors and residuals are computed with in floating-point arithmetic and are not modified.

FIG. 5.3. *Example 1 : exact, and blurred MIRE images*




**5.3.1. Example 1.** For the sake of showing the difficulty associated with noise, we first apply no noise, i.e., we take $\sigma = 0$. Figure 5.3 shows the original image, and the blurred image. The solution obtained by any linear solver, either a direct solver or Conjugate Gradient or PPF, is very accurate and is indistinguishible from the original image shown on the left of Figure 5.3, so it is omitted.

FIG. 5.4. *Example 2 : blurred noisy ($\sigma = 0.05$) MIRE image and image deblurred without regularization*



**5.3.2. Example 2.** Next, noise is added to the blurred image at a level of $\sigma = 0.05$. Figure 5.4 shows the blurred and noisy image and the deblurred image using a direct linear solver. The solution is clearly affected by noise and requires regularization.

FIG. 5.5. *Example 2 : PPF residuals and errors*

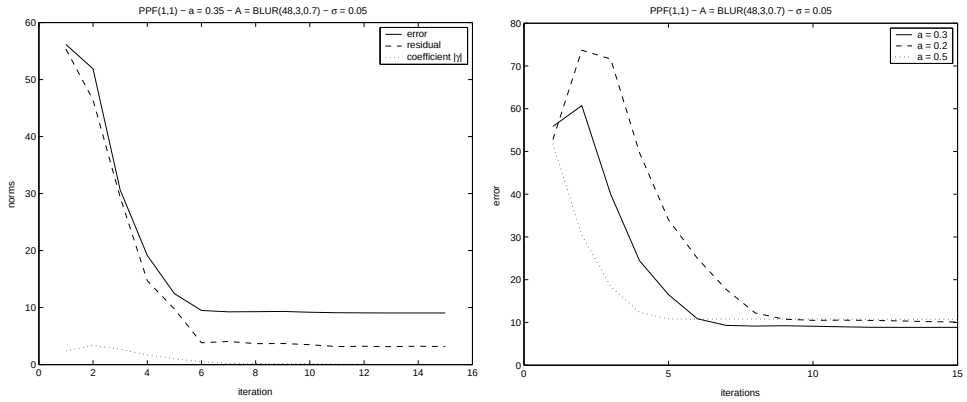
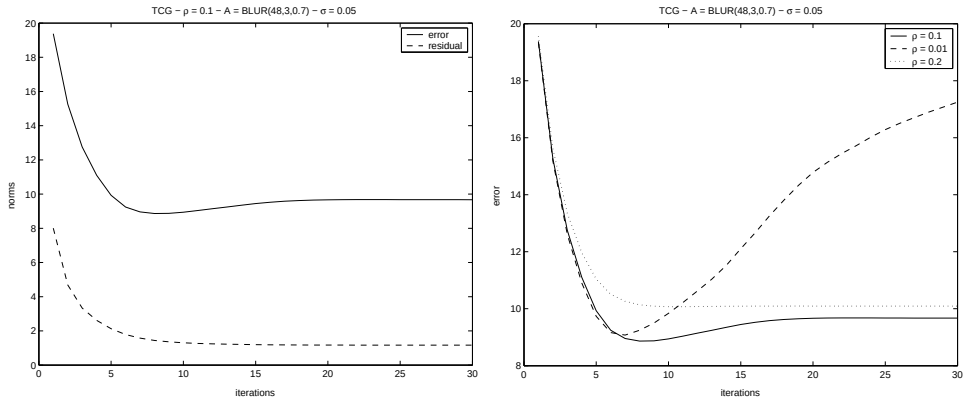
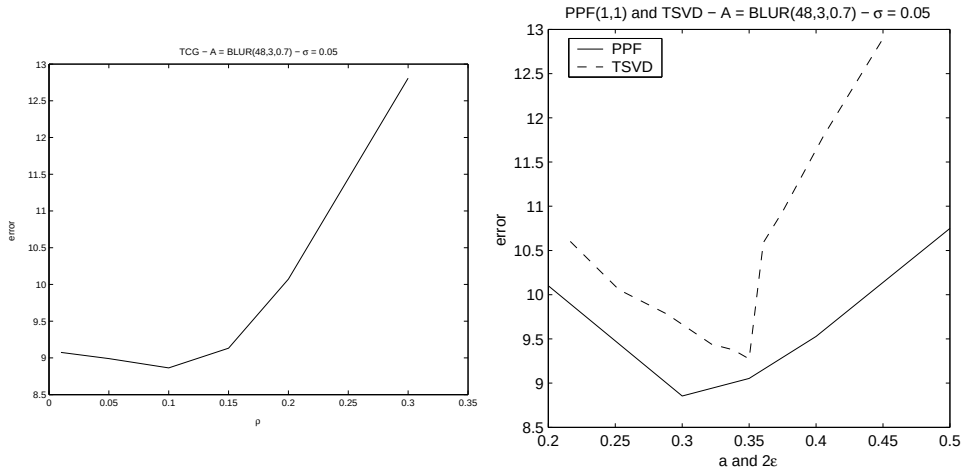



FIG. 5.6. *Example 2 : TCG residuals and errors*




We then used the PPF algorithm based on the filter $\phi^{[1,1]}$ with $a = 0.3$. Figure 5.5 shows the residual norm $\|b - Ax_k\|$, the error norm $\|x - x_k\|$ and the coefficients $\gamma_k$ with an increasing degree $k$. Here and in other cases as well, we observe a strong correlation between the three curves. This suggests that $\gamma_k$ could be exploited to obtain a stopping criterion. A decision on when to stop can be made in advance by generating the polynomials without computing the approximate solutions. The residual can also be used as a stopping criterion since both the residual and the error will stagnate when the corresponding polynomials have converged. In contrast, errors and residuals behave differently for Tychonov regularization

FIG. 5.7. *Example 2 : TCG, PPF and TSVD minimum errors*




coupled with a Conjugate Gradient method (TCG) as observed on Figure 5.6. Indeed, the residual continues to decrease while the error increases after a few iterations. Therefore, a good stopping criterion is required for TCG (Tychonov coupled with Conjugate Gradient), using the regularization properties of the Conjugate Gradient by itself.

A difficulty in regularization methods is to choose the best parameter, in some sense, either $\rho$ for TCG, $\epsilon$ for TSVD, or $a$ for PPF. Here we use as criterion the error norm and choose the parameter which minimizes this norm. For TCG and PPF, we also select the iteration which minimizes the error norm. Figure 5.7 plots the errors (the smallest ones reached during iteration for TCG and PPF) for the three methods, with varying parameter values. For TSVD, the abscissa plotted is $2\epsilon$. We observe that the parameter $a$ is about $2\epsilon$, which leads us to believe that the inflexion point $a/2$ is related to the parameter $\epsilon$.

FIG. 5.8. *Example 2 : MIRE images deblurred by TCG, PPF and TSVD*

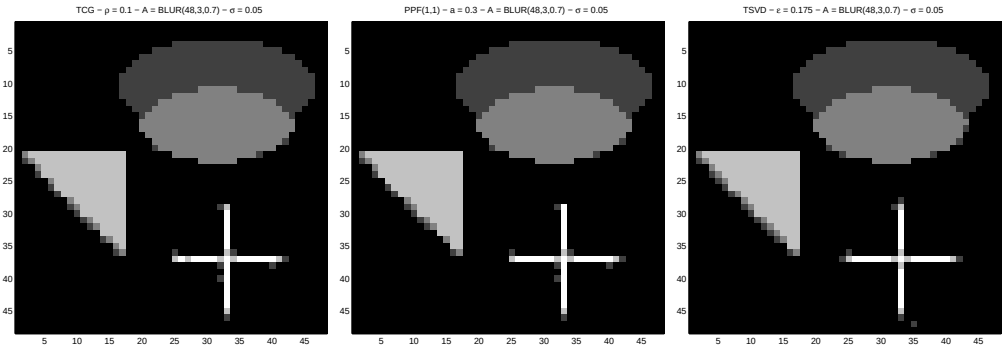
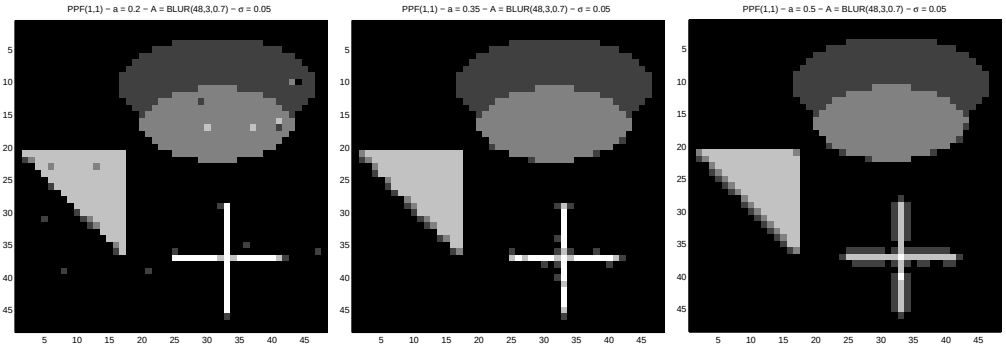


FIG. 5.9. *Example 2 : MIRE image deblurred by PPF* ($a = 0.2, a = 0.35, a = 0.5$)




In the three methods, we have selected the number of iterations and the parameter which minimize the error norm. Figure 5.8 shows the images deblurred, respectively, by TCG, PPF and TSVD, using these parameters. The quality of the three images is similar and the error is about the same, so PPF performs quite well on this example. To illustrate the effect of the regularizing parameter $a$, Figure 5.9 shows the images deblurred by PPF, with respectively $a = 0.2, 0.35, 0.5$. Clearly, a small value of $a$ adds too much noise while a large value of $a$ does not deblur sufficiently.

**5.3.3. Example 3.** Now we keep the same image and the same blurring matrix but we increase the noise to the level $\sigma = 0.15$. Figure 5.10 shows the blurred image and the image deblurred by a direct linear solver. Here, the deblurred image is dominated by noise.

FIG. 5.10. *Example 3 : blurred MIRE image with noise* ($\sigma = 0.15$) *and deblurred image without regularization*

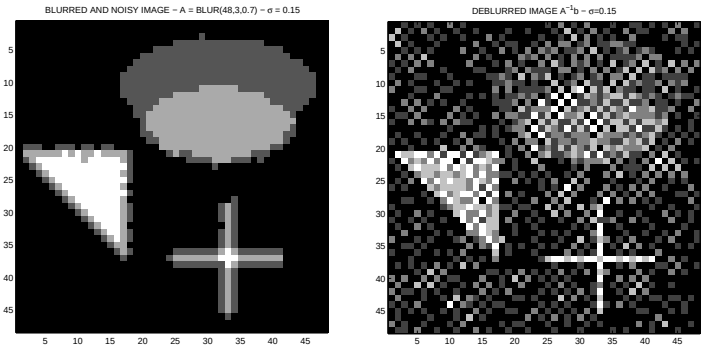



Figure 5.11 plots the error norms for methods TCG and TSVD, using the minimal error norm during iterations of TCG, with a varying parameter $\rho$ or $\epsilon$ (here, the abscissa is $\epsilon$). As expected, the parameter which minimizes the error norm is larger than in Example 2, because of the higher noise. The solid line in Figure 5.13 plots the error norm for PPF(1,1)

FIG. 5.11. *Example 3 : TCG and TSVD errors*

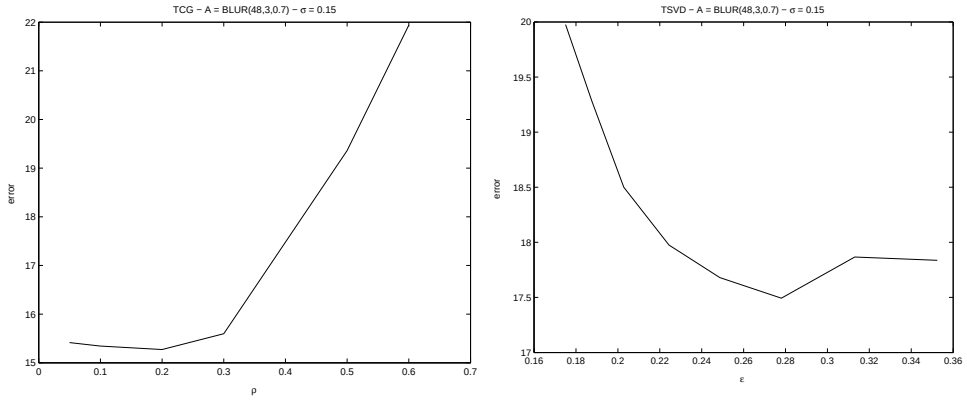
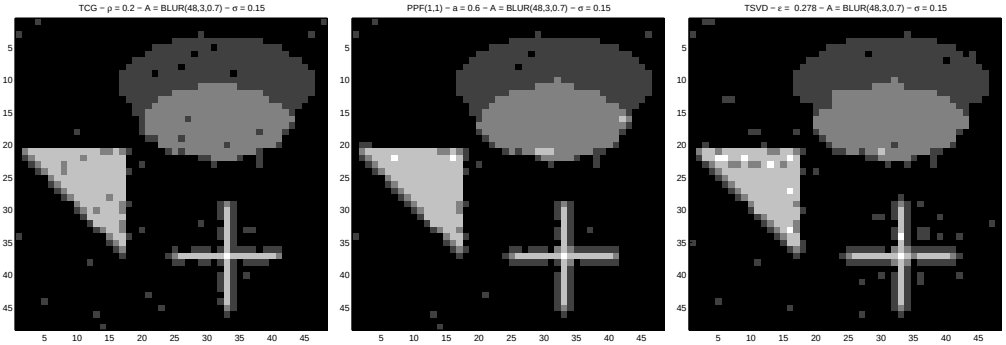


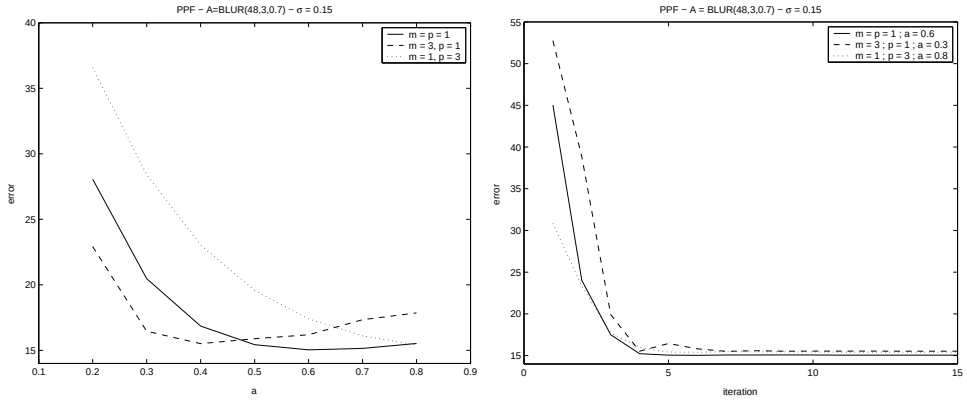

FIG. 5.12. *Example 3 : TCG, PPF and TSVD deblurred mire images*



against $a$, using also the minimal error norm during iterations. We still observe that the inflexion point $a/2 = 0.3$ for the best value $a$ is close to the best value $\epsilon = 0.28$. Figure 5.12 shows the images deblurred respectively by TCG, PPF and TSVD, using the parameters selected as previously on the basis of the minimum error.

**5.3.4. Example 4.** In this test, the matrix and the noise are the same as in Example 3. We now investigate the effect of the filter function, by varying the parameters $m, p$. Figure 5.13 plots the errors with $a$ varying for three choices : $(m = 1, p = 1), (m = 1, p = 3), (m = 3, p = 1)$. We observe that the value $a$ which minimizes the error is smaller when $p$ increases and larger when $m$ increases. For $(m = 1, p = 3)$, the best $a$ might not be in the interval $[0, 1]$. The errors for the best $a$ are about the same for the three filters. These parameters allow to ensure that the best $a$ will always be around 0.5 and that the method will always converge quickly.

FIG. 5.13. *Example 4 : PPF(1,1), PPF(3,1) and PPF(1,3) minimum errors*




**5.4. Example 5: a larger size image.** This image, referred to as EINSTEIN, is a photograph of Albert Einstein, of size 256 by 256, so that the matrix $A$ is of order 65536. Each pixel is coded on 256 integer grey levels, and we round again the pixels to integers in the interval $[0 : 255]$ when we plot the images (residuals and erros are computed with the floating-point numbers). The blurring matrix is $blur(256, 5, 0.7)$ and the noise is taken to $\sigma = 5$.

For PPF, the filter function is $\phi^{[1,1]}$. Here, TSVD can no longer be used due to the large size. Figure 5.14 shows the minimum errors with the method PPF for varying $a$ and the residuals history for the best value $a = 0.8$. As in previous examples, the residual and the error both stagnate after a few iterations. Figure 5.15 shows also the minimum errors with the method TCG for varying $\rho$ and the residuals history for the best value $\rho = 0.05$. The minimum error is slightly larger for TCG than for PPF. The better quality of PPF is confirmed by the images on Figure 5.16, which represents from left to right the EINSTEIN image deblurred by TCG, the blurred noisy EINSTEIN image and the EINSTEIN

FIG. 5.14. *Example 5 : PPF errors and residuals*

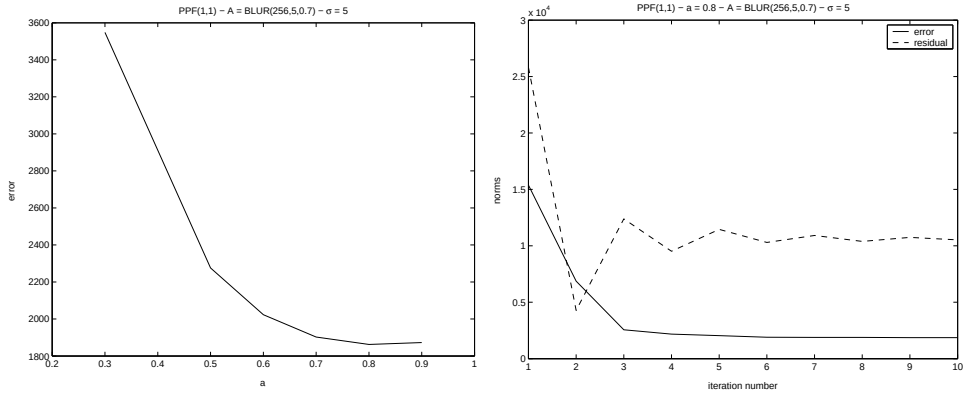



FIG. 5.15. *Example 5 : TCG errors and residuals*

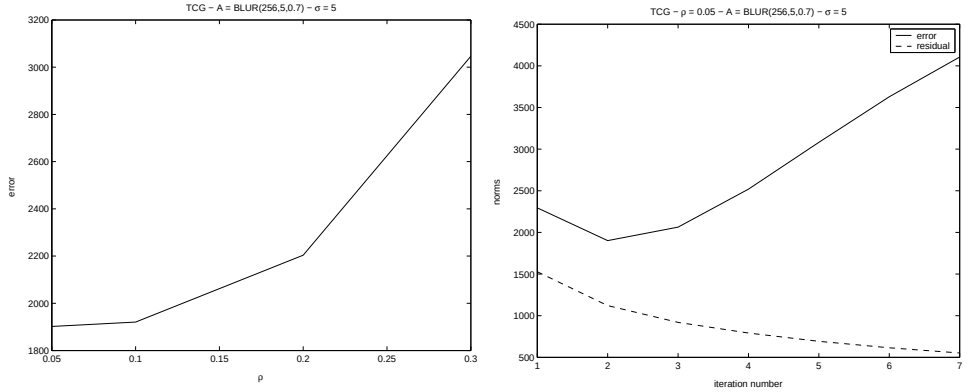



FIG. 5.16. *Example 5 : Einstein image - deblurred by TCG, blurred noisy, deblurred by PPF*

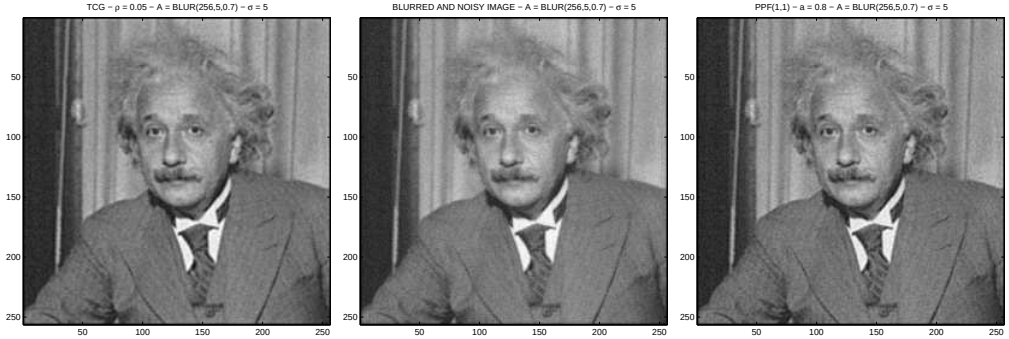


image deblurred by PPF. Regarding computational costs, these are roughly the same for both methods since one iteration of TCG requires two matrix-vector products whereas one iteration of PPF requires only one.

**6. Conclusion.** We have presented a method for regularizing ill-conditioned systems. The method consists of computing a polynomial approximation of a given "ideal" filter selected beforehand. The only restriction of the procedure is that this initial filter be piecewise polynomial.

The cost of one step of the algorithm is similar to that of one conjugate gradient iteration. We have shown that convergence of the iteration is at least of the order of $O(\frac{1}{\sqrt[3]{k}})$. While it seems that the approximate solution $x_k$ actually converges in practice the proof of convergence remains an open problem.

Though the experiments were carried out exclusively for image processing problems, the method is applicable to other applications where ill-conditioned systems arise. The numerical tests we have conducted in image recovery, show that the method behaves well, and one can see from the recovered images that the contours are better captured than with Tychonov regularization. Performance depends, however, on a careful choice of the parameter $a$. It should be possible to adapt well-knonwn techniques such as the L-curve or cross validation methods [3] for determining an optimal value of $a$.

REFERENCES


[1] D. Calvetti, L. Reichel, and Q. Zhang, *New iterative solution methods for large and very ill-conditioned linear systems of equations*, Numer. Math., (to appear).

[2] P. J. Davis, *Interpolation and Approximation*, Blaisdell, Waltham, MA, 1963.

[3] M. Hanke, *Conjugate gradient type methods for ill-posed problems*, Longman Scientific & Technical, Harlow, 1995.

[4] P. C. Hansen, *Truncated svd solutions to discrete ill-posed problems with ill-determined numerical rank*, Siam J. Sci. Statist. Comput., 11 (1990), pp. 503–518.

[5] ———, *Regularization tools.* A matlab Package for Analysis and Solution of Discrete Ill-Posed Problems, June 1992.

[6] P. C. Hansen, T. Sekii, and H. Shibahashi, *The modified truncated SVD method for regularization in general form*, Siam J. Sci. Statist. Comput., 13 (1992), pp. 1142–1150.

[7] Y. Huang, D. Kouri, and D. Hoffman, *Direct approaches to density functional theory: iterative treatment using polynomial representation of the Heaviside step function operator*, Chemical Physics Letters, 243 (1995), pp. 367–377.

[8] L. O. Jay, H. Kim, Y. Saad, and J. R. Chelikowsky, *Electronic structure calculations using plane wave codes without diagonlization*, Comput. Phys. Comm., 118 (1999), pp. 21–30.

[9] Y. Saad, *Iterative solution of indefinite symmetric systems by methods using orthogonal polynomials over two disjoint intervals*, SIAM Journal on Numerical Analysis, 20 (1983), pp. 784–811.

[10] ———, *Least squares polynomials in the complex plane and their use for solving sparse nonsymmetric linear systems*, SIAM Journal on Numerical Analysis, 24 (1987), pp. 155–169.

[11] A. N. Tikhonov, *Regularisation of incorrectly posed problems*, Soviet. Math. Dokl., 4 (1963), pp. 1624–1627.

[12] ———, *Solution of incorrectly formulated problems and the regularisation method*, Soviet. Math. Dokl., 4 (1963), pp. 1036–1038.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399